THE UNIVERSITY OF BRITISH COLUMBIA

Asymmetric Crypto System

EECE 412

Session 5

Last Session Recap

- Symmetric Cryptography
  - Classical cryptography
  - Sender, receiver share common key
  - Same key used for encryption and decryption
- Two basic types
  - Transposition (permutation) ciphers
  - Substitution ciphers
- Block cipher
  - SP-network
  - Modes of operation

Outline

- Asymmetric Cryptography
- Diffie Hellman
- RSA
- Cryptographic Checksums

Asymmetric Cryptography

- Public key cryptography
- Based on mathematical functions
- Two keys
  - Private key known only to individual (owner)
  - Public key available to anyone
- Idea
  - Confidentiality: encipher using public key, decipher using private key
  - Integrity/authentication: encipher using private key, decipher using public one

Requirements

- It must be computationally easy to encipher or decipher a message given the appropriate key
- It must be computationally infeasible to derive the private key from the public key
- It must be computationally infeasible to determine the private key from a chosen plaintext attack

Well-Known Public Key Schemes

- Diffie / Hellman (1976)
  - First public key scheme
  - Originally proposed for key exchange
- RSA (Rivest, Shamir, Adleman) (1977)
  - Only widely accepted and implemented general-purpose approach to public-key encryption.

Diffie-Hellman

- Compute a common, shared key
  - Called a *symmetric key exchange protocol*
- Based on discrete logarithm problem
  - Given integers n and g and prime number p, compute k such that $n = g^k \bmod p$
  - Solutions known for small p
  - Solutions computationally infeasible as p grows large
  - *{k} can be viewed as private key and {n} can be viewed as public key*

Algorithm

- Constants: prime p, integer $g \neq 0, 1, p-1$
  - Known to all participants
- Alice chooses private key *kAlice*, computes public key $KAlice = g^{kAlice} \bmod p$
- To communicate with Bob, Anne computes $Kshared = KBob^{kAlice} \bmod p$
- To communicate with Alice, Bob computes $Kshared = KAlice^{kBob} \bmod p$
  - It can be shown these keys are equal

Example

- Assume $p = 53$ and $g = 17$
- Alice chooses $kAlice = 5$
  - Then $KAlice = 17^5 \bmod 53 = 40$
- Bob chooses $kBob = 7$
  - Then $KBob = 17^7 \bmod 53 = 6$
- Shared key:
  - $KBob^{kAlice} \bmod p = 6^5 \bmod 53 = 38$
  - $KAlice^{kBob} \bmod p = 40^7 \bmod 53 = 38$

RSA

- Exponentiation cipher
- Relies on the difficulty of determining the number of numbers relatively prime to a large integer n
- Plaintext is encrypted in blocks m. (m < n)
- Encipher: $c = m^e \bmod n$
- Decipher: $m = c^d \bmod n$

$$(= (m^e)^d \bmod n$$
$$= m^{ed} \bmod n = m \bmod n)$$

- Sender knows: *{e, n} - public key*
- Receiver knows: *{d, n} - private key*

Requirements of RSA

- It is possible to find values of e, d, n such that $m = m^{ed} \bmod n$ for all $m < n$.
- It is relatively easy to calculate m^e and c^d for all values of m < n.
- It is infeasible to determine d given e and n.

Background

- Totient function $\phi(n)$
  - Number of positive integers less than *n* and relatively prime to *n*
    - *Relatively prime* means with no factors in common with *n*
- Example: $\phi(10) = 4$
  - 1, 3, 7, 9 are relatively prime to 10
- Example: $\phi(21) = 12$
  - 1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20 are relatively prime to 21

Algorithm

- Choose two large prime numbers p, q
  - Let $n = pq$; then $\phi(n) = (p-1)(q-1)$
  - Choose $e < n$ such that e is relatively prime to $\phi(n)$.
  - Compute d such that $ed \bmod \phi(n) = 1$
- Public key: (e, n); private key: (d, n);

  P, q, $\phi(n)$ are safely destroyed.
- Encipher: $c = m^e \bmod n$
- Decipher: $m = c^d \bmod n$

Example

- Take $p = 7$, $q = 11$, so $n = 77$ and $\phi(n) = 60$
- Alice chooses $e = 17$, making $d = 53$
- Bob wants to send Alice secret message HELLO (07 04 11 11 14)
  - $07^{17} \bmod 77 = 28$
  - $04^{17} \bmod 77 = 16$
  - $11^{17} \bmod 77 = 44$
  - $11^{17} \bmod 77 = 44$
  - $14^{17} \bmod 77 = 42$
- Bob sends 28 16 44 44 42

Example

- Alice receives 28 16 44 44 42
- Alice uses private key, $d = 53$, to decrypt message:
  - $28^{53} \bmod 77 = 07$
  - $16^{53} \bmod 77 = 04$
  - $44^{53} \bmod 77 = 11$
  - $44^{53} \bmod 77 = 11$
  - $42^{53} \bmod 77 = 14$
- Alice translates message to letters to read HELLO
  - No one else could read it, as only Alice knows her private key and that is needed for decryption

A few points

- It could be slow
  - Depend on the key length: 512, 1024, 2048, 4096…
- but . . .
  - I don’t have to distribute a secret key because I have my Private Key
  - Everyone with whom I communicate can know my Public Key
  - Scales well
    - There’s only one copy of the Private Key

Common Misconceptions

- It is more secure from cryptanalysis than is conventional encryption
- It makes conventional encryption obsolete
- Key distribution is trivial when using public-key encryption.

Security Services

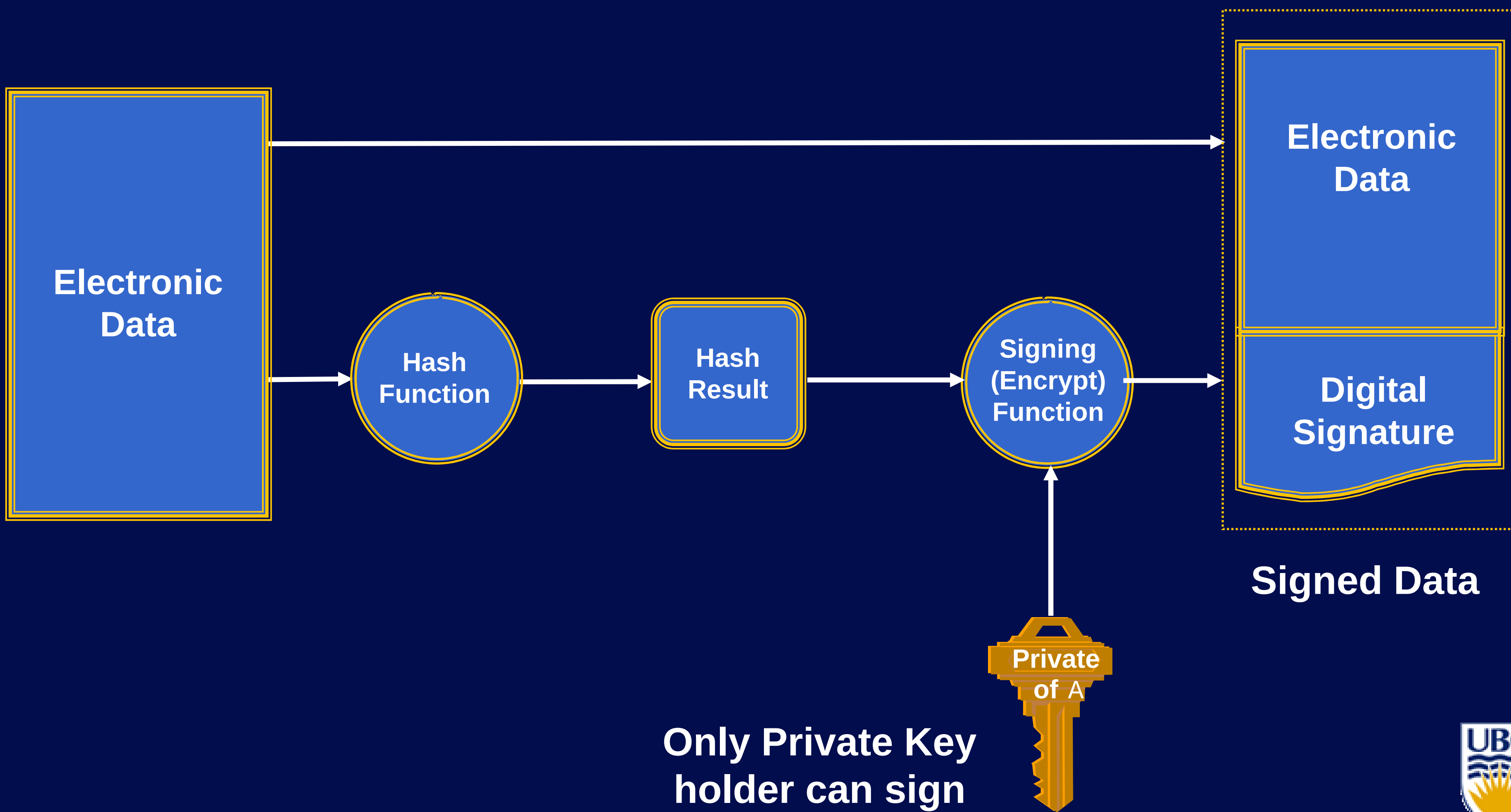
- Confidentiality
  - Only the owner of the private key knows it, so text enciphered with public key cannot be read by anyone except the owner of the private key
- Authentication
  - Only the owner of the private key knows it, so text enciphered with private key must have been generated by the owner

More Security Services

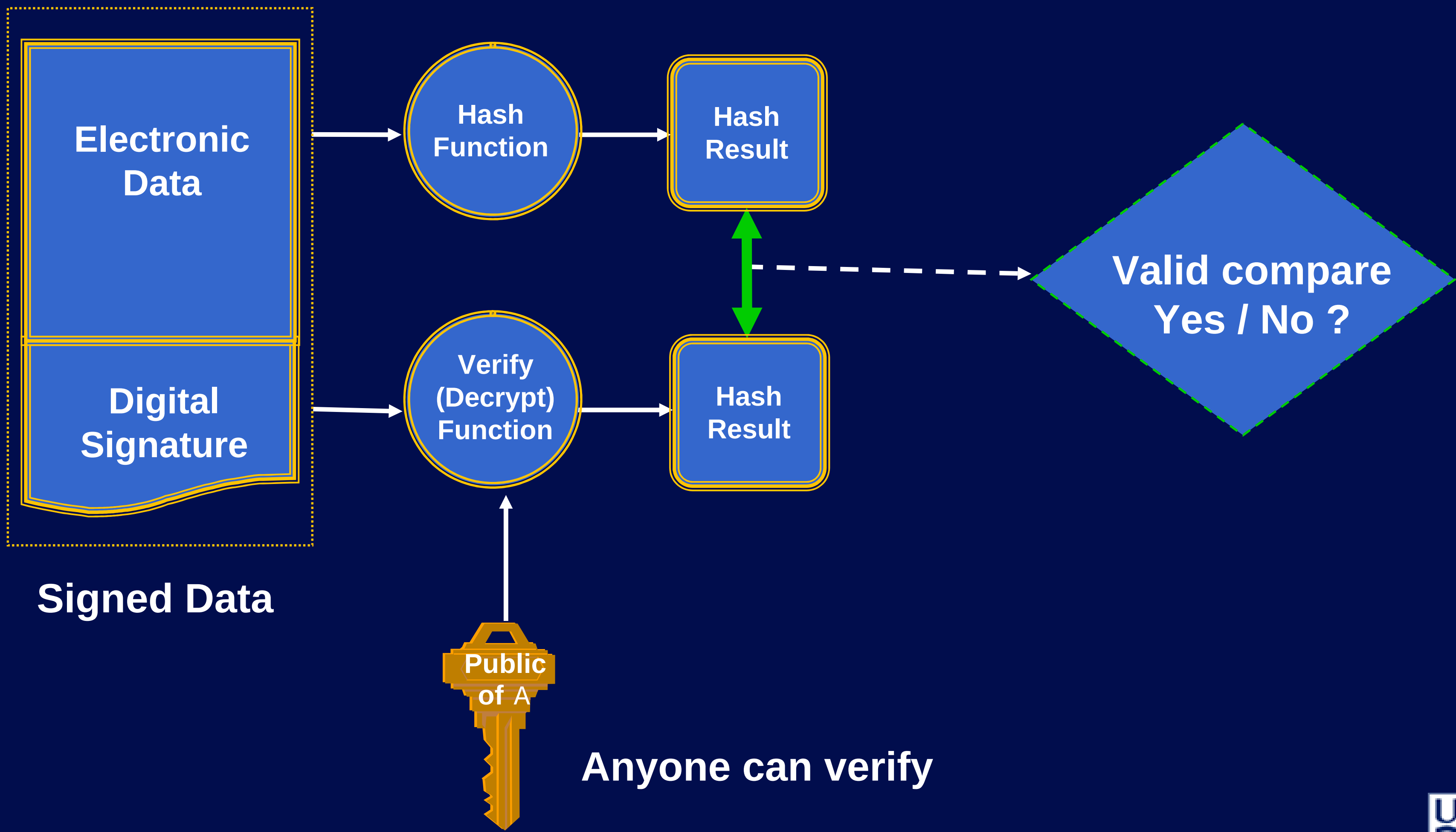
- Integrity
  - Enciphered letters cannot be changed undetectably without knowing private key
- Non-Repudiation
  - Message enciphered with private key came from someone who knew it

Digital Signature

- Authenticates sender's identity



Digital Signature Verification



Anyone can verify

Cryptographic Checksums

- Used for message authentication
- Mathematical function to generate a set of k bits from a set of n bits (where $k \leq n$).
  - k is smaller then n except in unusual circumstances
- Example: ASCII parity bit
  - ASCII has 7 bits; 8th bit is “parity”
  - Even parity: even number of 1 bits
  - Odd parity: odd number of 1 bits

Keys

- Keyed cryptographic checksum: requires cryptographic key
  - Message Authentication Code (MAC)
  - DES in chaining mode: encipher message, use last *n* bits. Requires a key to encipher, so it is a keyed cryptographic checksum.
- Keyless cryptographic checksum: requires no cryptographic key
  - Hash Function
  - MD5 and SHA-1 are best known; others include MD4, HAVAL, and Snefru

HMAC

- Incorporation of a secret key into an existing hash algorithm
- "A hash function such as MD5 was not designed for use as a MAC and cannot be used directly for that purpose because it does not rely on a secret key."

Algorithm

- h keyless cryptographic checksum function that takes data in blocks of b bytes and outputs blocks of l bytes. k' is cryptographic key of length b bytes
  - If short, pad with 0 bytes; if long, hash to length b
- $ipad$ is 00110110 repeated b times
- $opad$ is 01011100 repeated b times
- HMAC-$h(k, m) = h(k' \oplus opad \,||\, h(k' \oplus ipad \,||\, m))$
  - $\oplus$ exclusive or, $||$ concatenation

Key Points

- Public key cryptosystems encipher and decipher using different keys
  - Computationally infeasible to derive one from the other
- Cryptographic checksums provide a check on integrity
  - Keyed and Keyless

Next Session

- Lecture on Key Management
  - 50 minutes
- Quiz 1
  - 40 minutes